

Analysing Symbolic Regression Benchmarks under a Meta-Learning Approach

Luiz Otavio V. B. Oliveira, Joao Francisco B. S. Martins, Luis F. Miranda, Gisele L. Pappa

Universidade Federal de Minas Gerais, Department of Computer Science

Belo Horizonte, Brazil

[luizvbo,joaofbsm,luisfmiranda,glpappa]@dcc.ufmg.br

ABSTRACT

The definition of a concise and effective testbed for Genetic Programming (GP) is a recurrent matter in the research community. This paper takes a new step in this direction, proposing a different approach to measure the quality of the symbolic regression benchmarks quantitatively. The proposed approach is based on meta-learning and uses a set of dataset meta-features—such as the number of examples or output skewness—to describe the datasets. Our idea is to correlate these meta-features with the errors obtained by a GP method. These meta-features define a space of benchmarks that should, ideally, have datasets (points) covering different regions of the space. An initial analysis of 63 datasets showed that current benchmarks are concentrated in a small region of this benchmark space. We also found out that number of instances and output skewness are the most relevant meta-features to GP output error. Both conclusions can help define which datasets should compose an effective testbed for symbolic regression methods.

CCS CONCEPTS

•Computing methodologies → Genetic programming; Supervised learning by regression;

KEYWORDS

Genetic Programming; Benchmarks

ACM Reference format:

Luiz Otavio V. B. Oliveira, Joao Francisco B. S. Martins, Luis F. Miranda, Gisele L. Pappa. 2018. Analysing Symbolic Regression Benchmarks under a Meta-Learning Approach. In *Proceedings of Genetic and Evolutionary Computation Conference Companion, Kyoto, Japan, July 15–19, 2018 (GECCO '18 Companion)*, 8 pages.

DOI: 10.1145/3205651.3208293

1 INTRODUCTION

The quest for better Genetic Programming (GP) algorithms—e.g., capable of overcoming known drawbacks, presenting new interesting properties or operating with fewer resources—is as important as the search for better ways of evaluating these methods and comparing

them under different aspects. It is necessary to consistently and efficiently identify the scenarios where the new algorithm excels and how it compares to its predecessors. Otherwise, we may reach the wrong conclusions, which may mislead future studies and indicate an inappropriate method to solve the problem at hand.

The definition of a concise and effective testbed for GP is a recurrent matter in the research community. In the 14th Genetic and Evolutionary Computation Conference, McDermott et al. [18] brought to light the need for a redefinition of the datasets employed by the GP community. This work resulted in an extended study [36], which listed possible good and bad choices for benchmarking in GP. However, the quality of the benchmarks was measured subjectively using the feedback obtained from a GP community survey and the 14th GECCO attendees.

This work takes a different direction by analysing the datasets employed by the GP community using meta-features captured from the data. Our main idea is to give directions towards the proposal of an approach to measure the quality of the benchmarks quantitatively. Although previous works have already analysed GP datasets under a quantitative perspective [8, 27], to the best of our knowledge this is the first work to use meta-features. The objective of this paper is to characterize a set of GP benchmarks in symbolic regression according to the relationship of meta-features extracted from them and the error generated by GP.

In order to do that, we first identified the set of papers from GECCO 2013 to 2017 that dealt with symbolic regression. These papers used a set of 80 datasets. For 63 of them, we extracted a set of 11 meta-features, including, for instance, output skewness and the number of attributes, and ran a genetic programming (GP) method to obtain the normalized root mean square error (NRMSE). We then analysed the distribution of the values of these meta-features across the datasets according to the NRMSE obtained by GP, and plotted the “space of benchmarks” defined by these meta-features. The idea is to have dataset benchmarks that cover different regions of this space in order to assess whether new methods are really effective to solve the symbolic regression problem. The results showed that most of the datasets used as benchmarks in the literature present very similar meta-features and are concentrated in a small region of this benchmark space, and that output skewness and number of instances are the most relevant meta-features to predict the GP error.

The remainder of this paper is organized as follows. Section 2 discusses different approaches proposed to objectively assess the performance of learning methods, Section 3 presents the methodology adopted to characterize the datasets, while Section 4 analyses the experimental procedures conducted and their results. Finally,

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

GECCO '18 Companion, Kyoto, Japan

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM. 978-1-4503-5764-7/18/07...\$15.00

DOI: 10.1145/3205651.3208293

Section 5 draws conclusions about our findings and outlines future work directions.

2 RELATED WORK

As previously mentioned, the works in [18, 36] propose the first initiative to perform a critical analysis of the datasets used to benchmark genetic programming in general, using a somehow qualitative approach to define what a good benchmark is. Interestingly to say, many of the datasets suggested as blacklisted in [36] (e.g., quartic and lower order polynomials) were used in 14 out of 26 papers identified as dealing with symbolic regression in GECCO papers from 2013 to 2017.

Following these initial works, Nicolau et al. [27] analysed symbolic regression datasets presented by McDermott et al. [18], comparing different aspects of these benchmarks. They focused on synthetically generated datasets, exploring the sampling strategy and size and the addition of artificial noise to the data. The authors compared GP and Grammatical Evolution (GE) to two particularly simple baselines—a constant, corresponding to the average response observed in the training set, and a linear regression model. Results highlight a correlation between the distribution of the response variable and the performance of evolutionary algorithms. For datasets with a response variable with high variance, models induced by linear regression or by constant values were as good as or better than the ones induced by GP and GE.

In this same direction, Dick et al. [8] presented a quantitative analysis of the Human oral bioavailability dataset, used to guide the development of different GP-based methods. The authors analyse features with the assistance of information gain and Random Forests feature selection procedures, sampling variance and the performance of GP-based methods with Lasso, k -Nearest Neighbour and Random Forests for regression. The analysis shows that the dataset presents flaws that indicate it should be avoided in future works.

Although not applied to regression problems, Muñoz and colleagues [24, 25] present a framework to build a two-dimensional space based on meta-features of classification problems to identify regions where classification algorithms excel—and fail—and to generate datasets with the aim of enriching the diversity of classification problems in the meta-feature space.

Following Muñoz and colleagues [24, 25], the work presented here also explores the idea that the performance of algorithms can be visualized and pockets of the instance space corresponding to algorithm strengths and weaknesses can be identified. We extend that idea, however, to regression problems tackled by GP-based methods.

3 METHODOLOGY

As previously explained, the main objective of this paper is to characterize a set of GP benchmarks in symbolic regression according to the relationship of their features and the error generated by the GP. Recall that a symbolic regression task consists of inducing a model that maps inputs to outputs. More precisely, given a finite set of input-output pairs representing the training instances $T = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ —with $(\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$ and $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{id}]$, for $i = \{1, 2, \dots, n\}$ —we define $X = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]^T$ and $Y =$

$[y_1, y_2, \dots, y_n]^T$ as the matrix $n \times d$ of inputs and the n -element output vector, respectively. A symbolic regression consists then in inducing a model $p : X \rightarrow Y$ such that $\forall (\mathbf{x}_i, y_i) \in T : p(\mathbf{x}_i) = y_i$.

In order to characterize datasets used to benchmark methods to solve symbolic regression problems, we performed the following tasks:

- (1) We looked at the datasets that have been used to evaluate GP methods at GECCO papers from 2013 to 2017.
- (2) We extracted from these datasets a set of meta-features in order to characterize them. It is important to mention that the area of meta-learning has extensively studied attributes to characterize datasets, and this is still an open problem [3]. We have chosen 5 meta-features and generated them for both the training and test sets. In addition, we also considered the number of features.
- (3) We ran a canonical GP method in the selected datasets and recorded the normalized root mean squared error (NRMSE) for each dataset.
- (4) We created a meta-dataset, where each feature represents a meta-feature and is associated with the NRMSE generated for that respective dataset.
- (5) We used this dataset to determine which features were more related to the NRMSE obtained.

The next sections detail each of these steps.

3.1 Datasets

We started by compiling a list of 80 datasets employed in 26 papers working with symbolic regression problems published at GECCO from 2013 to 2017. From these 80 datasets, we could not find the description of two synthetic datasets—Sext [12] and Nguyen-12 [12, 16]—and we could not find seven real-world datasets available on line—Dow chemical [28], Plasma protein binding level (PPB) [5, 9, 29], Tower data [14, 15, 29], NOX [1, 2], Wind (WND) [15], Median lethal dose (toxicity/LD50) [5, 9] and Human oral bioavailability [5, 8, 9, 29]¹.

In addition, we decided not to use the *million song dataset*, given time restrictions, and Korns-2, Korns-3, Korns-5, Korns-6, Korns-8, Korns-9, Korns-10, given inconsistencies on the generated data—these datasets generate inputs that lead to inconsistent data, caused by division by zero, logarithm of zero or negative number and square root of negative number—ending up with 63 datasets. Tables 1 and 3 describe the real and the synthetic datasets, respectively. For the synthetic datasets, the training and test sets are sampled independently, according to two strategies. $U[a, b, c]$ indicates a uniform random sample of size c drawn from the interval $[a, b]$ and $E[a, b, c]$ indicates a grid of points evenly spaced with an interval c , from a to b , inclusive.

3.2 Meta-dataset predictive features

A set of six dataset-related meta-features were chosen, most of them with training and test equivalents, with exception of the number of features, which remains the same in all cases. They were:

- (1) Number of features;
- (2) Number of instances;

¹We intended to analyze only datasets freely available for download, in order to make the access to the data easier for the reader.

Table 1: Real datasets used in papers from GECCO 2013 to GECCO 2017.

Abbreviation	Dataset	# of features	# of instances	Source
ABA	Abalone ¹	8	500	[34]
AFN	Airfoil self-noise	6	1503	[29]
BOH	Boston housing	14	506	[8, 15, 34, 35]
CCP	Combined cycle power plant	4	9568	[20]
CPU	Computer hardware	9	209	[29]
CST	Concrete strength	9	1030	[20, 29]
ENC	Energy efficiency, cooling load	9	768	[1, 2, 15, 29]
ENH	Energy efficiency, heating load	9	768	[1, 2, 15, 29]
FFR	Forest fires	13	517	[29]
MSD	Million song dataset ²	90	1000000	[2]
OZO	Ozone ³	73	2536	[34]
WIR	Wine quality, red wine	12	1599	[1, 2, 29]
WIW	Wine quality, white wine	12	4898	[1, 2, 29]
YAC	Yacht hydrodynamics	7	308	[20]

¹ 500 randomly selected instances. The feature *sex* was represented as dummy variable.

² Dataset was not used due to execution time restrictions.

³ Missing values were replaced by the feature mean.

- (3) Target feature (output) skewness, defined in Eq. 1, where n is the number of instances, and $\bar{x}$ is the average of the sample;

$$Skewness = \frac{n\sqrt{n-1}}{n-2} \cdot \frac{\sum_{i=1}^n (x_i - \bar{x})^3}{[\sum_{i=1}^n (x_i - \bar{x})^2]^{3/2}} \quad (1)$$

- (4) Standard deviation of the target feature;
(5) Mean of the absolute feature-target correlation, calculated as the average of the Spearman correlation among each feature and the target output. The higher the value of this feature, the simpler the dataset;
(6) Linearity measure: In order to evaluate the “linearity” of a dataset, we used the coefficient of determination (R^2) of the model induced by a linear regression on the dataset. Recall that R^2 determines the percentage of variation in the output variable that can be explained by the linear relationship between the predictive features and the output. A low R^2 means there is not a strong linear relationship between predictive and output features.

We selected meta-features easily computed and directly related to the regression task. The number of instances reflect the available data for fitting and model evaluation; the number of features affect the input space; the skewness and standard deviation of the output capture the distribution of the target feature; the correlation feature-target captures the relation between the input and output features; and the linearity of the data is related to the shape of the function that generated the data.

It is important to mention that it makes sense to look at feature in both the training and test sets, as we are analyzing the quality of the datasets themselves, and not their generalization ability. In a future work, we intend to analyze the capability of modeling GP performance according to the meta-features.

3.3 Meta-dataset output feature

After extracting the meta-features from the 63 datasets, we ended up with a meta-dataset of 63 instances, each described by 11 features. The next step was to associate a metric of error to each of these instances. We adopted the Normalized Root Mean-Squared Error (NRMSE) [23] obtained by m runs of a GP as the meta-dataset output.

The NRMSE was chosen because, as we want to understand the relationship between the GP performance across different datasets, a normalized performance metric is more appropriate. We adopted the median over different executions of the GP method given by:

$$NRMSE = \frac{RMSE \cdot \sqrt{\frac{n}{n-1}}}{\sigma_Y} = \frac{\sqrt{\frac{\sum_{i=1}^n (y_i - f(x_i))^2}{n-1}}}{\sqrt{\frac{\sum_{i=1}^n (y_i - \bar{Y})^2}{n-1}}} \quad (2)$$

where $\bar{Y}$ and σ_Y are, respectively, the mean and standard deviation of the vector Y , composed by the expected outputs given by the training (or test) set, and f is the model (function) induced by the GP.

We defined different strategies for the GP experiments according to the nature and source of the datasets. For real datasets, we randomly partitioned the data into five disjoint sets of the same size and carried out the experiments six times with a 5-fold cross-validation (6×5 -CV). For the synthetic datasets, the data was sampled only once and the experiments were repeated 30 times with the same data.

We adopted the GP implementation from the *gplearn* Python package [32]. The fitness was defined as the NRMSE and the function set composed by $\{+, -, \times, AQ, sqrt, sin\}$, where the Analytic Quotient (AQ) [26] replaces the (protected) division and is defined as:

$$AQ(a, b) = \frac{a}{\sqrt{1 + b^2}}. \quad (3)$$

3.4 Meta-features Analyses

Having the meta-dataset defined, our final step was to perform a series of analyses to better understand the relations between the meta-features and the NRMSE. They were:

- (1) The meta-features relevance to determine the NRMSE, obtained according to a Random Forest Regressor (RFR). We adopted the implementation from Scikit-learn [30] to determine the importance of the features.
- (2) The meta-models generated using the meta-dataset created to predict the NRMSE of a GP when using a dataset with similar features. We adopted two approaches:
 - (a) We fitted a linear regression model to the meta-dataset, using the NRMSE as the response variable.
 - (b) We captured the variance of the meta-features by reducing the meta-dataset to two features, represented by the two principal components generated by the Principal Component Analysis (PCA) method [11]—applied after the meta-features were normalized. Then we fitted a linear regression to this two-dimensional data, using the NRMSE as the response variable. This strategy allows us to visualize the meta-features (the components representing them), along with the NRMSE, in a three-dimensional plot.

4 EXPERIMENTAL ANALYSIS

This section shows the results of the analyses described in the previous section². Note that the output values were obtained by running a canonical GP framework with a population of 1,000 individuals evolved for 50 generations. Subtree crossover, subtree mutation, hoist mutation and point mutation were applied with probabilities 0.85, 0.05, 0.05 and 0.05, respectively, and the tournament selection had size 10. All other parameters were set to their default value.

4.1 Meta-dataset analysis

Intuitively, a good set of benchmarks would include problems with a great variety of characteristics. Our methodology characterizes datasets according to a set of 11 meta-features, which we believe are relevant to show the relationship between the problem and the performance of GP, but they are far from being a complete set. Here we analyze the distribution of the values of these meta-features across the 63 datasets considered, and the results are illustrated in Figure 1. A general observation across all graphs is that there is a concentration of datasets in one point of this “space” that illustrates the benchmarks. For example, looking at the number of features (Figure 1a), we observed that the great majority of the datasets has one or two features, with the maximum number being 13. The same happens for the number of instances in the training and test sets (Figures 1b and 1e), with the majority of datasets having less than 2,000 instances. Of course, that time restrictions need to be considered when creating benchmarks, but a few datasets with a

Table 2: Meta-feature importance according to the Random Forest Regressor. The higher the value the more important is the feature.

Meta-Feature	Feat. import.
Target skewness (Training)	0.290170
N. instances (Training)	0.184041
N. features	0.122526
Linearity measure (Training)	0.075096
Target skewness (Test)	0.056585
Target std (Training)	0.054541
Linearity measure (Test)	0.052174
Target std (Test)	0.051961
Mean absolute correlation attribute-target (Test)	0.044588
Mean absolute correlation attribute-target (Training)	0.037164
N. instances (Test)	0.031154

greater number of examples can be beneficial to the analysis of the proposed method.

The mean absolute correlation of the predictive features and the target output (showed in Figures 1c and 1f) shows that, individually, the features present, on average, very low correlation with the output. However, there is still 12 datasets with average correlation greater than 0.8 in the training set—*Keijzer-6*, *Keijzer-7*, *Keijzer-8*, *Keijzer-9*, *Nguyen-1*, *Nguyen-2*, *Nguyen-3*, *Nguyen-6*, *Nguyen-7*, *Nguyen-8*, *R1* and *R2*—and 13 datasets in the test set—*Keijzer-6*, *Keijzer-7*, *Keijzer-8*, *Keijzer-9*, *Nguyen-1*, *Nguyen-2*, *Nguyen-3*, *Nguyen-4*, *Nguyen-6*, *Nguyen-7*, *Nguyen-8*, *Nonic*, *R1* and *R2*. As expected, also observe in these figures that the real datasets present lower average correlation than the synthetic ones.

Regarding our linearity measure represented by R^2 (Figures 1g and 1j), observe that most datasets (46% for the training and 49% for the test set) have a R^2 lower than 0.5, which may be an indication that most of them are not described by linear relationships among the predictive features and the output. Finally, looking at the target skewness (Figures 1h and 1k), notice that most of the datasets are concentrated from -1 to 1, with a peak around 0. A skewness value of 0 (meaning the data are perfectly symmetrical) is quite unlikely for real-world data, although we have seven real-world datasets—for both training and test set—with skewness in the interval [-0.5, 0.5]. As a rule of thumb, datasets with skewness greater(smaller) than 1(-1) are considered highly skewed. We have 23(21) highly and 11(9) moderate skewed (with values between -1 and -0.5 and 0.5 and 1) training(test) datasets.

4.2 Meta-features relevance

In order to analyse the importance of each meta-feature on the prediction of GP performance—measured by the median NRMSE on the test set—we fitted the Random Forest Regressor from Scikit-learn [30] to the meta-dataset—composed of the meta-features of all datasets as input and the respective median NRMSE as output. Table 2 presents the feature importance inferred by the Random Forest Regressor from Scikit-learn [30]. The number of trees in the forest was set to 120, according to a parameter tuning using the randomized search from Scikit-learn—all the other parameters were set to the default value. Notice that the target skewness and

²The code used in our experimental analysis is freely available for download on GitHub at <https://github.com/laic-ufmg/gp-benchmarks>

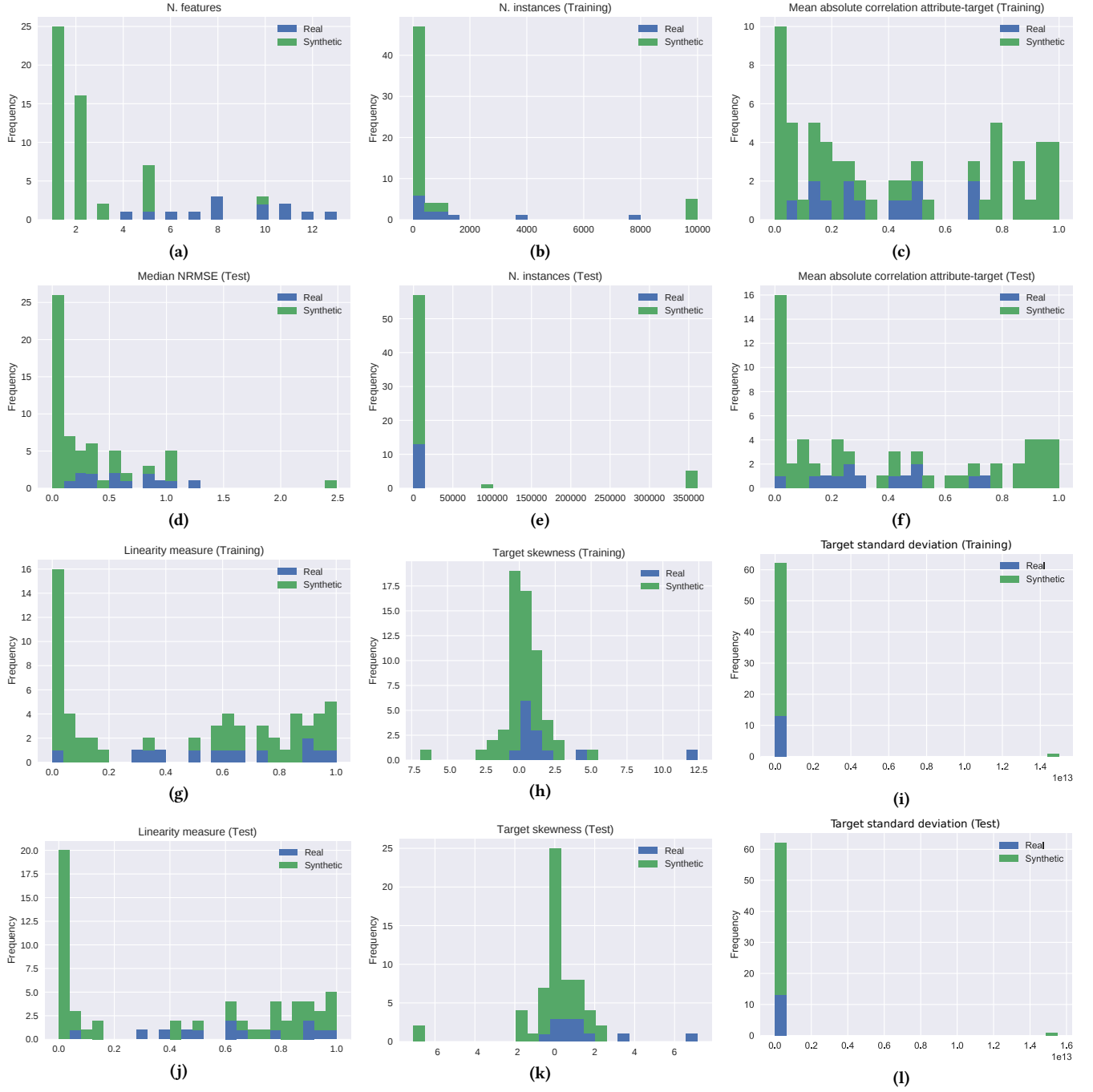


Figure 1: Distribution of the values of the datasets meta-features across the 63 datasets considered.

the number of instances in the training set were considered the most relevant meta-features, followed by the number of features.

Figure 2 presents the values of NRMSE in the test set according to the two most important features according to Table 2. Circles correspond to real-world datasets and triangles to synthetic ones. The colour of each element and its position on the vertical axis

represent the median test NRMSE obtained as output by the regression models. Note that the concentration of datasets with skewness close to 0 leads to very small errors in these datasets, which are mostly synthetic ones. For the number of instances, we observe that smaller datasets tend to have smaller error, especially for synthetic datasets—with the exception of two datasets, *Korns-1* and *Korns-4*, which present small NRMSE and have 10,000 training instances.

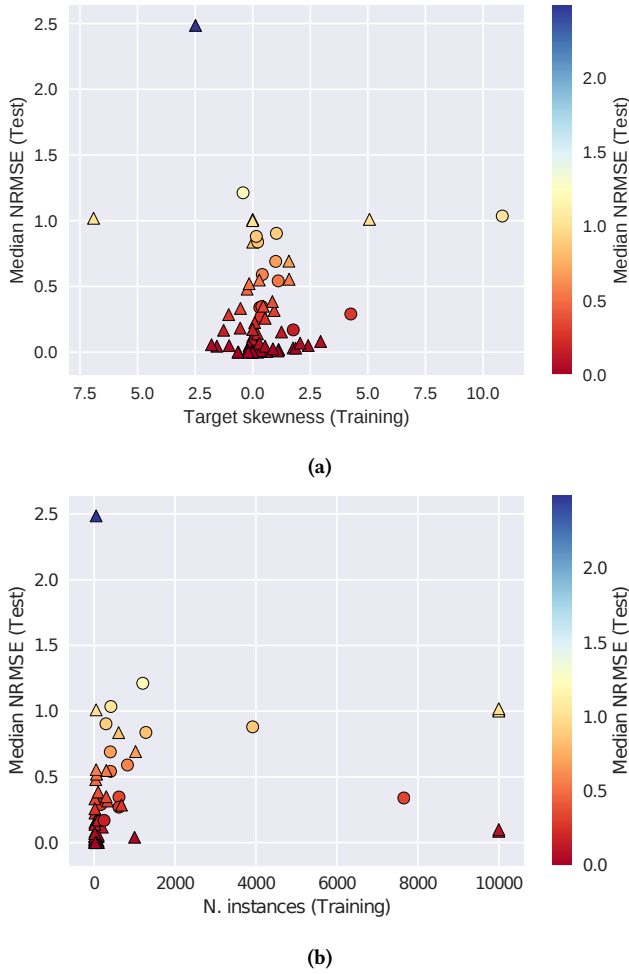


Figure 2: Relation between the (a) the target skewness in the training set and (b) the number of instances in the training set compared to the median test NRMSE. Circles represent real datasets, while triangles represent synthetic ones.

In order to be able to visualize where the datasets fall in the dataset space generated by the meta-features, we plotted in Figure 3 the two principal components of PCA—induced from the meta-features normalized to $[0, 1]$ —together with the test NRMSE returned by GP. Again, circles correspond to real-world datasets and triangles to synthetic ones, and the colour of each element and its position on the vertical axis represent the median test NRMSE obtained as output by the GP. The figure also shows the least-squares plane generated using a linear regression method. Observe that all datasets, with a few exceptions, are really concentrated in a portion of the space generated by these two components. In an ideal scenario of benchmarks, there should be a better distribution of these datasets in this space. The plane generated using these two components as a representation for the meta-features obtained a coefficient of determination of 0.332 and a RMSE of 0.356.

If the same linear regression is applied to the full set of meta-features, the coefficient of determination increases to 0.491 and the

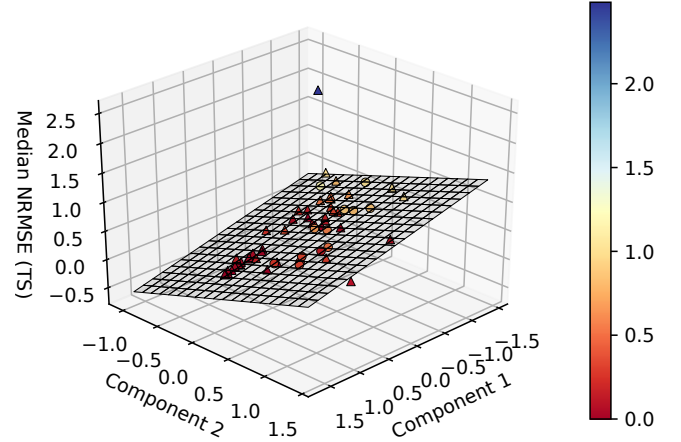


Figure 3: Relation between the median test NRMSE and meta-features mapped to a two-dimensional space using the PCA method. Circles represent real datasets, while triangles represent synthetic ones.

RMSE decreases to 0.311. However, the Random Forest Regressor model, that generated the feature importance previously reported, had a coefficient of determination of 0.894 and a RMSE of 0.142, giving evidence that the relations among the predictive features and the target output are mostly non-linear.

5 CONCLUSIONS AND FUTURE WORK

The main contribution of this paper was to move the GP community towards the proposal of a framework to measure the quality of the benchmarks quantitatively. We started by performing an analysis of the datasets commonly used as benchmarks for symbolic regression in the GP scenario by using a meta-learning approach. This approach extracted a set of meta-features from the benchmark datasets, and analysed how they correlate to the output error of a canonical GP. We extracted 11 meta-features from 63 datasets, analysed how the later were distributed according to the former and also reduced these 11 meta-features to two using a dimensionality reduction method. In this way, we were able to visualize the meta-feature space and observe that most examples were concentrated in the same region of this induced space. This suggests that the datasets are still very similar, and more variety is desired in a set of benchmarks.

As future work, we want to first explore a wider set of meta-features, and add new datasets to our meta-dataset. We also intend to analyse different regression methods to build an improved model for the NRMSE prediction from meta-features.

ACKNOWLEDGMENTS

This work was partially supported by the following Brazilian Research Support Agencies: CNPq, FAPEMIG, CAPES. It has also been partially funded by the EUBra-BIGSEA project by the European Commission under the Cooperation Programme (MCTI/RNP 3rd Coordinated Call), Horizon 2020 grant agreement 690116.

REFERENCES

- [1] Ignacio Arnaldo, Krzysztof Krawiec, and Una-May O'Reilly. 2014. Multiple regression genetic programming. In *GECCO '14: Proceedings of the 2014 conference on Genetic and evolutionary computation*. ACM, New York, NY, USA, 879–886.
- [2] Ignacio Arnaldo, Una-May O'Reilly, and Kalyan Veeramachaneni. 2015. Building Predictive Models via Feature Synthesis. In *GECCO '15: Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 983–990.
- [3] Pavel Brazdil, Christophe Giraud Carrier, Carlos Soares, and Ricardo Vilalta. 2008. *Metalearning: Applications to data mining*. Springer Science & Business Media.
- [4] Armand R. Burks and William F. Punch. 2015. An Efficient Structural Diversity Technique for Genetic Programming. In *GECCO '15: Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 991–998.
- [5] Mauro Castelli, Leonardo Trujillo, Leonardo Vanneschi, Sara Silva, Emigdio Z-Flores, and Pierrick Legrand. 2015. Geometric Semantic Genetic Programming with Local Search. In *GECCO '15: Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 999–1006.
- [6] Qi Chen, Bing Xue, Lin Shang, and Mengjie Zhang. 2016. Improving Generalisation of Genetic Programming for Symbolic Regression with Structural Risk Minimisation. In *GECCO '16: Proceedings of the Genetic and Evolutionary Computation Conference 2016*. ACM, New York, NY, USA, 709–716. DOI: <http://dx.doi.org/10.1145/2908812.2908842>
- [7] Vinicius Veloso De Melo. 2014. Kaizen programming. In *GECCO '14: Proceedings of the 2014 conference on Genetic and evolutionary computation*. ACM, New York, NY, USA, 895–902. DOI: <http://dx.doi.org/10.1145/2576768.2598264>
- [8] Grant Dick, Aysha P. Rimoni, and Peter A. Whigham. 2015. A Re-Examination of the Use of Genetic Programming on the Oral Bioavailability Problem. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation (GECCO '15)*. ACM, New York, NY, USA, 1015–1022. DOI: <http://dx.doi.org/10.1145/2739480.2754771>
- [9] Ivo Gonçalves, Sara Silva, Carlos M. Fonseca, and Mauro Castelli. 2017. Unsure when to stop?: ask your semantic neighbors. In *GECCO '17: Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 929–936. DOI: <http://dx.doi.org/10.1145/3071178.3071328>
- [10] Tomohiro Harada and Keiki Takadama. 2014. Asynchronously evolving solutions with excessively different evaluation time by reference-based evaluation. In *GECCO '14: Proceedings of the 2014 conference on Genetic and evolutionary computation*. ACM, New York, NY, USA, 911–918.
- [11] H. Hotelling. 1933. Analysis of a complex of statistical variables into principal components. *J. Educ. Psych.* 24 (1933).
- [12] Krzysztof Krawiec and Una-May O'Reilly. 2014. Behavioral programming: a broader and more detailed take on semantic GP. In *GECCO '14: Proceedings of the 2014 conference on Genetic and evolutionary computation*. ACM, New York, NY, USA, 935–942. DOI: <http://dx.doi.org/10.1145/2576768.2598288>
- [13] Krzysztof Krawiec and Tomasz Pawlak. 2013. Approximating geometric crossover by semantic backpropagation. In *GECCO '13: Proceeding of the fifteenth annual conference on Genetic and evolutionary computation*. ACM, New York, NY, USA, 941–948. DOI: <http://dx.doi.org/10.1145/2463372.2463483>
- [14] William La Cava, Thomas Helmuth, Lee Spector, and Kourosh Danai. 2015. Genetic Programming with Epigenetic Local Search. In *GECCO '15: Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 1055–1062. DOI: <http://dx.doi.org/10.1145/2739480.2754763>
- [15] William La Cava, Lee Spector, and Kourosh Danai. 2016. Epsilon-Lexicase Selection for Regression. In *GECCO '16: Proceedings of the Genetic and Evolutionary Computation Conference 2016*. ACM, New York, NY, USA, 741–748. DOI: <http://dx.doi.org/10.1145/2908812.2908898>
- [16] Paweł Liskowski and Krzysztof Krawiec. 2017. Discovery of search objectives in continuous domains. In *GECCO '17: Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 969–976. DOI: <http://dx.doi.org/10.1145/3071178.3071344>
- [17] Rui L. Lopes and Ernesto Costa. 2013. GEARNet: grammatical evolution with artificial regulatory networks. In *GECCO '13: Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*. ACM, New York, NY, USA, 973–980. DOI: <http://dx.doi.org/10.1145/2463372.2463490>
- [18] James McDermott, David R. White, Sean Luke, Luca Manzoni, Mauro Castelli, Leonardo Vanneschi, Wojciech Jaskowski, Krzysztof Krawiec, Robin Harper, Kenneth De Jong, and Una-May O'Reilly. 2012. Genetic programming needs better benchmarks. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*. ACM, 791–798.
- [19] Nicholas Freitag McPhee, M. Kirbie Dramdahl, and David Donatucci. 2015. Impact of Crossover Bias in Genetic Programming. In *GECCO '15: Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 1079–1086. DOI: <http://dx.doi.org/10.1145/2739480.2754778>
- [20] David Medernach, Jeannie Fitzgerald, R. Muhammad Atif Azad, and Conor Ryan. 2016. A New Wave: A Dynamic Approach to Genetic Programming. In *GECCO '16: Proceedings of the Genetic and Evolutionary Computation Conference 2016*. ACM, New York, NY, USA, 757–764.
- [21] Eric Medvet, Fabio Daolio, and Danny Tagliapietra. 2017. Evolvability in grammatical evolution. In *GECCO '17: Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 977–984. DOI: <http://dx.doi.org/10.1145/3071178.3071298>
- [22] Andreas Meier, Mark Gonter, and Rudolf Kruse. 2013. Accelerating convergence in cartesian genetic programming by using a new genetic operator. In *GECCO '13: Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*. ACM, New York, NY, USA, 981–988. DOI: <http://dx.doi.org/10.1145/2463372.2463481>
- [23] Luis F. Miranda, Luiz Otavio V. B. Oliveira, Joao Francisco B. S. Martins, and Gisele L. Pappa. 2017. How noisy data affects geometric semantic genetic programming. In *GECCO '17: Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 985–992. DOI: <http://dx.doi.org/10.1145/3071178.3071300>
- [24] Mario A Muñoz and Kate Smith-Miles. 2017. Generating custom classification datasets by targeting the instance space. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 1582–1588.
- [25] Mario A Muñoz, Laura Villanova, Davaatseren Baatar, and Kate Smith-Miles. 2018. Instance spaces for machine learning classification. *Machine Learning* 107, 1 (2018), 109–147.
- [26] Ji Ni, Russ H Driberg, and Peter I Rockett. 2013. The use of an analytic quotient operator in genetic programming. *IEEE Transactions on Evolutionary Computation* 17, 1 (2013), 146–152.
- [27] Miguel Nicolau, Alexandros Agapitos, Michael O'Neill, and Anthony Brabazon. 2015. Guidelines for defining benchmark problems in genetic programming. In *Evolutionary Computation (CEC), 2015 IEEE Congress on*. IEEE, 1152–1159.
- [28] Miguel Nicolau and Michael Fenton. 2016. Managing Repetition in Grammar-Based Genetic Programming. In *GECCO '16: Proceedings of the Genetic and Evolutionary Computation Conference 2016*. ACM, New York, NY, USA, 765–772. DOI: <http://dx.doi.org/10.1145/2908812.2908904>
- [29] Luiz Otavio V.B. Oliveira, Fernando E.B. Otero, and Gisele L. Pappa. 2016. A Dispersion Operator for Geometric Semantic Genetic Programming. In *GECCO '16: Proceedings of the Genetic and Evolutionary Computation Conference 2016*. ACM, New York, NY, USA, 773–780. DOI: <http://dx.doi.org/10.1145/2908812.2908923>
- [30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
- [31] Léo Francisco Dal Piccol Sotto and Vinicius Veloso de Melo. 2017. A probabilistic linear genetic programming with stochastic context-free grammar for solving symbolic regression problems. In *GECCO '17: Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 1017–1024.
- [32] Trevor Stephens. 2018. gplearn: Genetic Programming in Python. (2018). <https://github.com/trevorstephens/gplearn> [Online; accessed 30/03/2018].
- [33] Marcin Szubert, Anuradha Kodali, Sangram Ganguly, Kamalika Das, and Josh C. Bongard. 2016. Reducing Antagonism between Behavioral Diversity and Fitness in Semantic Genetic Programming. In *GECCO '16: Proceedings of the Genetic and Evolutionary Computation Conference 2016*. ACM, New York, NY, USA, 797–804. DOI: <http://dx.doi.org/10.1145/2908812.2908939>
- [34] Pham Thi Thuong, Nguyen Xuan Hoai, and Xin Yao. 2017. Combining conformal prediction and genetic programming for symbolic interval regression. In *GECCO '17: Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 1001–1008. DOI: <http://dx.doi.org/10.1145/3071178.3071280>
- [35] Peter A. Whigham, Grant Dick, James Maclaurin, and Caitlin A. Owen. 2015. Examining the "Best of Both Worlds" of Grammatical Evolution. In *GECCO '15: Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*. ACM, New York, NY, USA, 1111–1118. DOI: <http://dx.doi.org/10.1145/2739480.2754784>
- [36] David R White, James McDermott, Mauro Castelli, Luca Manzoni, Brian W Goldman, Gabriel Kronberger, Wojciech Jaskowski, Una-May O'Reilly, and Sean Luke. 2013. Better GP benchmarks: community survey results and proposals. *Genetic Programming and Evolvable Machines* 14, 1 (March 2013), 3–29. DOI: <http://dx.doi.org/10.1007/s10710-012-9177-2>
- [37] Bartosz Wieloch and Krzysztof Krawiec. 2013. Running programs backwards: instruction inversion for effective search in semantic spaces. In *GECCO '13: Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*. ACM, New York, NY, USA, 1013–1020. DOI: <http://dx.doi.org/10.1145/2463372.2463493>
- [38] Tony Worm and Kenneth Chiu. 2013. Prioritized grammar enumeration: symbolic regression by dynamic programming. In *GECCO '13: Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*. ACM, New York, NY, USA, 1021–1028. DOI: <http://dx.doi.org/10.1145/2463372.2463486>

Table 3: Synthetic datasets used in GECCO papers from 2013-2017.

Dataset	Variables	Objective Function	Training Set	Testing Set	Source
Burks (*)	1	$4 * x_1^4 + 3 * x_1^3 + 2 * x_1^2 + x_1$	$U[-1, 1, 20]$	None	[4, 33]
Keijzer-1	1	$0.3 * x^1 * \sin(2 * \pi * x_1)$	$E[-1, 1, 0.1]$	$E[-1, 1, 0.001]$	[7, 12, 13, 16, 23]
Keijzer-2	1	$0.3 * x^1 * \sin(2 * \pi * x_1)$	$E[-2, 2, 0.1]$	$E[-2, 2, 0.001]$	[7, 23]
Keijzer-3	1	$0.3 * x^1 * \sin(2 * \pi * x_1)$	$E[-3, 3, 0.1]$	$E[-3, 3, 0.001]$	[7, 23]
Keijzer-4	1	$x_1^3 * e^{-x_1} * \cos(x_1) * \sin(x_1) * (\sin^2(x_1) * \cos(x_1) - 1)$	$E[0, 10, 0.1]$	$E[0.05, 10.05, 0.1]$	[6, 7, 12, 13, 16, 23, 31, 33]
Keijzer-5	3	$\frac{30 * x_1 * x_3}{(x_1 - 10) * x_2^2}$	$x_1, x_2 : U[-1, 1, 1000]$ $x_3 : U[1, 2, 1000]$	$x_1, x_2 : U[-1, 1, 10000]$ $x_3 : U[1, 2, 10000]$	[7, 12, 29, 31]
Keijzer-6	1	$\sum_{i=1}^{x_1} i$	$E[1, 50, 1]$	$E[1, 120, 1]$	[7, 14, 21, 23, 28, 29]
Keijzer-7	1	$\ln x_1$	$E[1, 100, 1]$	$E[1, 100, 0.1]$	[7, 23, 29]
Keijzer-8	1	$\sqrt{x_1}$	$E[0, 100, 1]$	$E[0, 100, 0.1]$	[7, 16, 23]
Keijzer-9	1	$\operatorname{arcsinh}(x_1) = \ln(x_1 + \sqrt{x_1^2 + 1})$	$E[0, 100, 1]$	$E[0, 100, 0.1]$	[7, 23]
Keijzer-10	2	$x_1^{x_2}$	$U[0, 1, 100]$	$E[0, 1, 0.01]$	[7, 34, 37]
Keijzer-11	2	$x_1 * x_2 + \sin((x_1 - 1) * (x_2 - 1))$	$U[-3, 3, 20]$	$E[-3, 3, 0.01]$	[6, 7, 12, 34]
Keijzer-12 (*)	2	$x_1^4 - x_1^3 + (\frac{x_2}{2}) - x_2$	$U[-3, 3, 20]$	$E[-3, 3, 0.01]$	[6, 7, 12, 34, 37]
Keijzer-13	2	$6 * \sin(x_1) * \cos(x_2)$	$U[-3, 3, 20]$	$E[-3, 3, 0.01]$	[7, 12, 34]
Keijzer-14	2	$\frac{8}{2 + x_1^2 + x_2^2}$	$U[-3, 3, 20]$	$E[-3, 3, 0.01]$	[6, 7, 12, 16, 34]
Keijzer-15	2	$\frac{x_1}{5} + \frac{x_2}{2} - x_2 - x_1$	$U[-3, 3, 20]$	$E[-3, 3, 0.01]$	[6, 7, 12, 16, 34]
Korns-1	1	$1.57 + 24.3 * x_4$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[38]
Korns-2 ¹	3	$0.23 + 14.2 * \frac{x_4 + x_2}{3 * x_5}$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[38]
Korns-3 ¹	4	$-5.41 + 4.9 * \frac{x_4 - x_1 + \frac{x_2}{x_5}}{3 * x_5}$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[31, 38]
Korns-4	1	$-2.3 + 0.13 * \sin(x_3)$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[38]
Korns-5 ¹	1	$3 + 2.13 * \ln(x_5)$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[31, 38]
Korns-6 ¹	1	$1.3 + 0.13 * \sqrt{x_1}$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[38]
Korns-7	1	$213.80940889 * (1 - e^{-0.54723748542 * x_1})$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[38]
Korns-8 ¹	3	$6.87 + 11 * \sqrt{7.23 * x_1 * x_4 * x_5}$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[38]
Korns-9 ¹	4	$\frac{\sqrt{x_1}}{\ln(x_2)} * \frac{e^{x_3}}{x_4^2}$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[38]
Korns-10 ¹	4	$0.81 + 24.3 * \frac{2 * x_2 + 3 * x_3^2}{4 * x_4^3 + 5 * x_5^4}$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[38]
Korns-11	1	$6.87 + 11 * \cos(7.23 * x_1^3)$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[38]
Korns-12	2	$2 - 2.1 * \cos(9.8 * x_1) * \sin(1.3 * x_5)$	$U[-50, 50, 10000]$	$U[-50, 50, 10000]$	[38]
Koza-2 (*)	1	$x_1^5 - 2 * x_1^3 + x_1$	$U(-1, 1, 20)$	None	[22]
Koza-3 (*)	1	$x_1^6 - 2 * x_1^4 + x_1^2$	$U(-1, 1, 20)$	None	[10, 22]
Meier-3	2	$\frac{x_1^2 * x_2^2}{x_1 + x_2}$	$U[-1, 1, 50]$	None	[22]
Meier-4	2	$\frac{x_1}{x_2^3}$	$U[-1, 1, 50]$	None	[22]
Nguyen-1 (*)	1	$x_1^3 + x_1^2 + x_1$	$U(-1, 1, 20)$	None	[7, 31, 38]
Nguyen-2 (*)	1	$x_1^4 + x_1^3 + x_1^2 + x_1$	$U(-1, 1, 20)$	None	[7, 10, 17, 21, 31, 35, 38]
Nguyen-3	1	$x_1^5 + x_1^4 + x_1^3 + x_1^2 + x_1$	$U(-1, 1, 20)$	None	[7, 12, 16, 31, 37, 38]
Nguyen-4	1	$x_1^6 + x_1^5 + x_1^4 + x_1^3 + x_1^2 + x_1$	$U(-1, 1, 20)$	None	[7, 12, 16, 31, 37, 38]
Nguyen-5	1	$\sin(x_1^2) * \cos(x_1) - 1$	$U(-1, 1, 20)$	None	[7, 10, 12, 16, 37, 38]
Nguyen-6	1	$\sin(x_1) + \sin(x_1 + x_1^2)$	$U(-1, 1, 20)$	None	[7, 12, 16, 31, 37, 38]
Nguyen-7	1	$\ln(x_1 + 1) + \ln(x_1^2 + 1)$	$U[0, 2, 20]$	None	[13], [7, 10, 12, 14, 16, 37, 38]
Nguyen-8	1	$\sqrt{x_1}$	$U[0, 4, 20]$	None	[7, 16, 37, 38]
Nguyen-9	2	$\sin(x_1) + \sin(x_2^2)$	$U(-1, 1, 100)$	None	[7, 12, 16, 37, 38]
Nguyen-10	2	$2 * \sin(x_1) * \cos(x_2)$	$U(-1, 1, 100)$	None	[7, 12, 37, 38]
Nonic	1	$\sum_{i=1}^9 x_i^4$	$E[-1, 1, 20]$	$U[-1, 1, 20]$	[13, 33]
Pagie-1	2	$\frac{1}{1 + x_1^{-4}} + \frac{1}{1 + x_2^{-4}}$	$E[-5, 5, 0.4]$	None	[14, 16, 19]
Poly-10	9	$x_1 * x_2 + x_3 * x_4 + x_5 * x_6 + x_1 * x_7 * x_9 + x_3 * x_6 * x_{10}$	$U[0, 1, 330]$	$U[0, 1, 170]$	[20]
R1	1	$\frac{(x_1 + 1)^3}{x_1^2 - x_1 + 1}$	$E[-1, 1, 20]$	$U[-1, 1, 20]$	[13, 16, 33]
R2	1	$\frac{x_1^5 - 3 * x_1^3 + 1}{x_1^2 + 1}$	$E[-1, 1, 20]$	$U[-1, 1, 20]$	[13, 16, 33]
R3	1	$\frac{x_1^6 + x_1^5}{x_1^4 + x_1^3 + x_1^2 + x_1 + 1}$	$E[-1, 1, 20]$	$U[-1, 1, 20]$	[16]
Sine	1	$\sin(x_1) + \sin(x_1 + x_1^2)$	$E[0, 6.2, 0.1]$	None	[19]
Vladislavleva-1	2	$\frac{e^{-(x_1 - 1)^2}}{1.2 + (x_2 - 2.5)^2}$	$U[0.3, 4, 100]$	$E[-0.2, 4.2, 0.1]$	[16, 23, 29, 34]
Vladislavleva-2	1	$e^{-x_1} * x_1^2 * (\cos x_1 * \sin x_1) * (\cos x_1 * \sin^2 x_1 - 1)$	$E[0.05, 10, 0.1]$	$E[-0.5, 10.5, 0.05]$	[23]
Vladislavleva-3	2	$e^{-x_1} * x_1^3 * (\cos x_1 * \sin x_1) * (\cos x_1 * \sin^2 x_1 - 1) * (x_2 - 5)$	$x_1 : E[0.05, 10, 0.1]$ $x_2 : E[0.05, 10.05, 2]$	$x_1 : E[-0.5, 10.5, 0.05]$ $x_2 : E[-0.5, 10.5, 0.5]$	[23]
Vladislavleva-4	5	$\frac{10}{5 + \sum_{i=1}^5 (x_i - 3)^2}$	$U[0.05, 6.05, 1024]$	$U[-0.25, 6.35, 5000]$	[14, 15, 20, 28, 29]
Vladislavleva-5	3	$30 * (x_1 - 1) * \frac{x_3 - 1}{(x_1 - 10) * x_2^2}$	$x_1 : U[0.05, 2, 300]$ $x_2 : U[1, 2, 300]$ $x_3 : U[0.05, 2, 300]$	$x_1 : E[-0.05, 2.1, 0.15]$ $x_2 : E[0.95, 2.05, 0.1]$ $x_3 : E[-0.05, 2.1, 0.15]$	[6, 34]
Vladislavleva-6	2	$6 * \sin(x_1) * \cos(x_2)$	$U[0.1, 5.9, 30]$	$E[-0.05, 6.05, 0.02]$	[6, 34]
Vladislavleva-7	2	$(x_1 - 3) * (x_2 - 3) + 2 * \sin((x_1 - 4) * (x_2 - 4))$	$U[0.05, 6.05, 300]$	$U[-0.25, 6.35, 1000]$	[23]
Vladislavleva-8	2	$\frac{(x_1 - 3)^4 + (x_2 - 3)^3 - (x_2 - 3)}{(x_2 - 2)^4 + 10}$	$U[0.05, 6.05, 50]$	$E[-0.25, 6.35, 0.2]$	[6, 34]

¹ Note that the domain may include values for which the function is undefined.

(*) Datasets not recommended as benchmarks in [36].